

```
1 <?php
2
3 /**
4  * @author GOLAY Brian
5  * @version 1.0 (2021/05/20)
6  * Group-related functions
7  */
8
9 namespace FlightClub\sql;
10
11 use FlightClub\sql\DBConnection;
12
13 require_once 'dbConnection.php';
14
15 /**
16  * Group related class that contain group related functions with database
17  * interaction
18  */
19 class GroupDAO
20 {
21     /**
22      * return all pending invites
23      *
24      * @return array[mixed] pending invites data
25      */
26     public static function getAllPendingInvites()
27     {
28         $db = DBConnection::getConnection();
29         $sql = "SELECT * FROM `tbl_user-group` WHERE `Dttm_Membership` is null";
30         $request = $db->prepare($sql);
31         $request->execute();
32         return $request->fetchAll();
33     }
34
35     /**
36      * return all group data with the group Id
37      *
38      * @param int $idGroup
39      * @return array[mixed] all group data
40      */
41     public static function getGroupById($idGroup)
42     {
43         $db = DBConnection::getConnection();
44         $sql = "SELECT * FROM `tbl_group` WHERE `Id_Group` = :id";
45         $request = $db->prepare($sql);
46         $request->execute([
47             ':id' => $idGroup
48         ]);
49         return $request->fetch();
50     }
51
52     /**
53      * Delete a pending invite
54      *
55      * @param int $idGroup
56      * @param int $idUser
57      * @return void
58      */
59     public static function deletePendingInvite($idGroup, $idUser)
60     {
61     }
```

```

60         $db = DBConnection::getConnection();
61         $sql = "DELETE FROM `tbl_user-group` WHERE `Id_Group` = :idGroup AND
`Id_User` = :idUser";
62
63         $request = $db->prepare($sql);
64         $request->execute([
65             ":idGroup" => $idGroup,
66             ":idUser" => $idUser
67         ]);
68     }
69
70     /**
71      * Accept a pending invite
72      *
73      * @param int $idGroup
74      * @param int $idUser
75      * @return void
76      */
77     public static function acceptPendingInvite($idGroup, $idUser)
78     {
79         $dateTime = date("Y-m-d H:i:s");
80         $db = DBConnection::getConnection();
81         $sql = "UPDATE `tbl_user-group` SET `Dttm_Membership` = :date WHERE
`tbl_user-group`.`Id_User` = :idUser AND `tbl_user-group`.`Id_Group` = :idGroup;";
82
83         $request = $db->prepare($sql);
84         $request->execute([
85             ":idGroup" => $idGroup,
86             ":idUser" => $idUser,
87             ":date" => $dateTime
88         ]);
89     }
90
91     /**
92      * Get all groups of one user identified by his id
93      *
94      * @param int $id
95      * @return array[mixed]
96      */
97     public static function getAllOfMyGroup($id)
98     {
99         $db = DBConnection::getConnection();
100         $sql = "SELECT * FROM `tbl_user-group` join `tbl_group` on `tbl_user-
group`.`Id_Group` = `tbl_group`.`Id_Group` WHERE `Dttm_Membership` is not null AND
`tbl_user-group`.`Id_User` = :id";
101         $request = $db->prepare($sql);
102         $request->execute([
103             ':id' => $id
104         ]);
105         return $request->fetchAll();
106     }
107
108     /**
109      * Get all users of one group identified by the group id
110      *
111      * @param int $idGroup
112      * @return array[mixed]
113      */
114     public static function getAllOfGroupUser($idGroup)
115     {

```

```

116         $db = DBConnection::getConnection();
117         $sql = "SELECT * FROM `tbl_user-group` join `tbl_user` on `tbl_user-
group`.`Id_User` = `tbl_user`.`Id_User` WHERE `Dttm_Membership` is not null AND
`tbl_user-group`.`Id_Group` = :id";
118         $request = $db->prepare($sql);
119         $request->execute([
120             ':id' => $idGroup
121         ]);
122         return $request->fetchAll();
123     }
124
125     /**
126      * Get an users of one group identified by his id
127      *
128      * @param int $idGroup
129      * @param int $idUser
130      * @return array[mixed]
131      */
132     public static function getGroupUserByIdUserAndIdGroup($idUser, $idGroup)
133     {
134         $db = DBConnection::getConnection();
135         $sql = "SELECT * FROM `tbl_user-group` join `tbl_user` on `tbl_user-
group`.`Id_User` = `tbl_user`.`Id_User` WHERE `Dttm_Membership` is not null AND
`tbl_user-group`.`Id_Group` = :idGroup AND `tbl_user-group`.`Id_User` = :idUser";
136         $request = $db->prepare($sql);
137         $request->execute([
138             ':idUser' => $idUser,
139             ':idGroup' => $idGroup
140         ]);
141         return $request->fetchAll();
142     }
143
144     /**
145      * Create a group
146      *
147      * @param string $name
148      * @return void
149      */
150     public static function createGroup($name)
151     {
152         $db = DBConnection::getConnection();
153         $sql = "INSERT INTO `tbl_group` (`Id_Group`, `Nm_Group`) VALUES (NULL,
:name);";
154
155         $request = $db->prepare($sql);
156         $request->execute([
157             ":name" => $name
158         ]);
159     }
160
161     /**
162      * return all group data sort by the highest group id descending
163      *
164      * @return array[mixed]
165      */
166     public static function getLastGroupId()
167     {
168         $db = DBConnection::getConnection();
169         $sql = "SELECT * FROM `tbl_group` order by `Id_Group` desc";
170         $request = $db->prepare($sql);

```

```
171     $request->execute();
172     return $request->fetchAll();
173 }
174
175 /**
176  * Join a group just created by the user
177  *
178  * @param int $idGroup
179  * @param int $idUser
180  * @return void
181  */
182 public static function joinCreatedGroup($idGroup, $idUser)
183 {
184     $dateTime = date("Y-m-d H:i:s");
185     $db = DBConnection::getConnection();
186     $sql = "INSERT INTO `tbl_user-group` (`Id_User`, `Id_Group`,
`Dttm_Invitation`, `Dttm_Membership`) VALUES (:idUser, :idGroup, :date, :date)";
187
188     $request = $db->prepare($sql);
189     $request->execute([
190         ":idGroup" => $idGroup,
191         ":idUser" => $idUser,
192         ":date" => $dateTime
193     ]);
194 }
195
196
197 /**
198  * Create and send an invitation to an user
199  *
200  * @param int $idGroup
201  * @param int $idUser
202  * @return void
203  */
204 public static function CreateInvite($idGroup, $idUser)
205 {
206     $dateTime = date("Y-m-d H:i:s");
207     $db = DBConnection::getConnection();
208     $sql = "INSERT INTO `tbl_user-group` (`Id_User`, `Id_Group`,
`Dttm_Invitation`, `Dttm_Membership`) VALUES (:idUser, :idGroup, :date, null)";
209
210     $request = $db->prepare($sql);
211     $request->execute([
212         ":idGroup" => $idGroup,
213         ":idUser" => $idUser,
214         ":date" => $dateTime
215     ]);
216 }
217
218
219 /**
220  * return the user group data with the group Id and user id
221  *
222  * @param int $idGroup
223  * @return array[mixed] all group data
224  */
225 public static function getMyGroupById($idGroup, $idUser)
226 {
227     $db = DBConnection::getConnection();
```

```
228         $sql = "SELECT * FROM `tbl_user-group` WHERE `Id_Group` = :idGroup AND
`Id_User` = :idUser AND Dttm_Membership is not null";
229         $request = $db->prepare($sql);
230         $request->execute([
231             ':idGroup' => $idGroup,
232             ':idUser' => $idUser
233         ]);
234         return $request->fetch();
235     }
236 }
237
```